MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

RAČUNALSTVO

IVAN LESJAK

Sustav za praćenje i upravljanje temperaturom u radnom prostoru

ZAVRŠNI RAD

ČAKOVEC, 2017

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
RAČUNALSTVO

IVAN LESJAK

System for monitoring and controlling temperature in the workplace

ZAVRŠNI RAD

Mentor:
mr. sc. Željko Knok

ČAKOVEC, 2017

Zahvala

Zahvaljujem se svojoj obitelji na strpljenju i podršci kod izrade ovog završnog rada i za pomoć kroz sve godine studiranja. Zahvaljujem se također mentoru Željku Knoku na pruženoj prilici i na pomoći i savjetima.

Najiskrenija hvala svim kolegama i prijateljima koji su na bilo koji način pomogli u izradi ovog rada.

SAŽETAK

U ovom završnom radu bit će riječi o mogućnostima programibilnog mikrokontrolera zvanog „Arduino“, o senzorima i modulima za „Arduino“,o bazama podataka te o C# programskom jeziku.

Cilj ovog završnog rada je istražiti mogućnosti „Arduina“ te pokušati napraviti što vjerniju simulaciju sigurnosnog sustava za neko radno postrojenje. U završnom radu je korištena pločica“Arduino Uno“.

Konceptualno je zamišljeno da se na „Arduino“ spoje određeni senzori poput temperaturnog senzora „DHT11“ koji mjeri temperaturu u stupnjevima Celzijevim i vlagu zraka u postocima, senzora čistoće zraka „MQ135“ koji iščitava u zraku alkoholne pare, benzene i amonijak te modul detektora zvuka koji određuje stupanj buke u prostoriji. „Arduino“ bi podatke trebao skupiti te proslijediti računalu koje će te podatke preuzeti i spremiti u bazu podataka napravljenu u Microsoft Visual Studio pomoću tehnologije „Entity Framework“ koja omogućuje rad s relacijskim bazama podataka korištenjem Microsoft .NET specifičnih jezika. Tu se nalaze i LED diode za svaki senzor koje služe kao indikator potencijalno opasnih situacija te da se prikaže koji senzor je u području prihvatljivih vrijednosti, a koji nije.

Neke od prednosti „Arduina“ u izradi ovog završnog rada bile su njegova cijena, jednostavnost korištenja, veličina, veliki izbor senzora i modula te načina spajanja s drugim uređajima, dok su nedostaci ograničenost pinovima (iako je potencijalno rješenje nabava veće pločice, odnosno pločice koja ima više pinova poput „Arduino Mega“), te sam editor koda koji je limitiran dok njegove biblioteke troše puno RAM memorije koje ga usporavaju u radu. „Arduino“ i baza podataka spojeni su preko USB kabla jer limitiranost „Arduina“ nije dopuštala 'Bluetooth' vezu čime bi se postigla mobilnost završnog rada. Zbog nepreglednosti žica i općenitog vizualnog dojma, završni rad je smješten u plastično kućište.

Više detalja o korištenim metodama, senzorima i tehnologijama moguće je pronaći u nastavku teksta koji je popraćen slikama i referencama u nadi da će pomoći drugim studentima ili krajnjim korisnicima koji se odluče za sličan rad.

Ključne riječi: *Arduino, Entity Framework, baze podataka, moduli, senzori*

SADRŽAJ

1. UVOD

Nadzor i upravljanje nad prostorijama osim PLC-evima i SCADA sustavom moguće je izvesti i pomoću Arduino pločice. PLC (*Programmable Logic Controller*) je industrijsko kontrolno računalo koje neprekidno nadgleda stanja ulaznih uređaja i odlučuje pomoću prilagođenog programa kontrolu stanja izlaznih uređaja. Skoro svaka produkcijska linija, funkcija uređaja ili proces može biti poboljšan pomoću ovog tipa kontrolnog sustava. Najveća korist u korištenju PLC-a je mogućnost promjene ili ponavljanje postupka ili procesa dok se prikupljaju i vitalne informacije. Još jedna prednost PLC sustava je u tome da je modularan, odnosno pruža mogućnosti miješanja i spajanja različitih tipova ulaznih i izlaznih uređaja koji vam najbolje odgovaraju. Glavna razlika između računala i PLC-a je tome što su PLC-evi napravljeni da rade u težim radnim uvjetima (poput onih koji uključuju prisustvo veće količine prašine, vlage, topline ili hladnoće) te imaju mogućnost širih ulazno/izlaznih aranžmana. Oni spajaju PLC sa senzorima i pogonima. PLC-evi upravljaju električnim motorima, pneumatskim i hidrauličnim cilindrima, magnetskim relejima, zavojnicama ili analognim izlazima. PLC programi su većinom pisani u posebnim aplikacijama ili na osobnim računalima te tada skinuti direktnom vezom ili preko mreže na PLC.[4][5]

SCADA (*Supervisory control and data acquisition*) je industrijski automatizirani kontrolni sustav koji je jezgra mnogih modernih industrija koje uključuju: energiju, naftu i plin, vodovod i otpadne vode, recikliranje i slično. SCADA sustave koriste privatne kompanije i javni sektori. SCADA radi dobro u različitim tipovima poduzeća jer ima širok opseg konfiguracija, od jednostavnijih do kompleksnijih projekata. Praktički se svugdje u današnjem svijetu koristi neka inačica SCADA sustava, bila to lokalna trgovina, rafinerija, otpadne vode ili vlastiti dom. SCADA sustavi koriste više strojne opreme i programa koji omogućavaju praćenje, sakupljanje i procesuiranje podataka, interakciju i kontrolu nad strojevima i uređajima poput ventila, pumpi, motora i ostalog koji su spojeni preko HMI (ljudsko-strojnog sučelja) programa. U osnovi SCADA arhitekture, informacije se iz senzora ili ručnih unosa šalju na PLC ili RTU (*remote terminal units*) koji onda šalju informacije računalima sa SCADA programima.

SCADA programi analiziraju i prikazuju podatke tako da pomognu operatorima i drugim radnicima da smanje otpad i poboljšaju učinkovitost u procesu proizvodnje.[6]

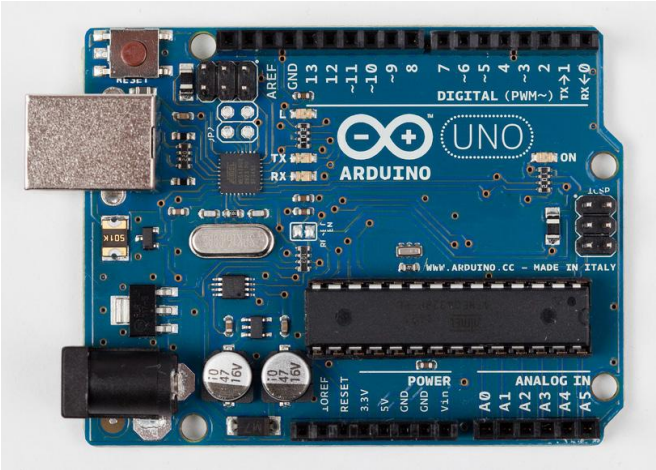
2. SKLOPOVLJE

2.1. ARDUINO

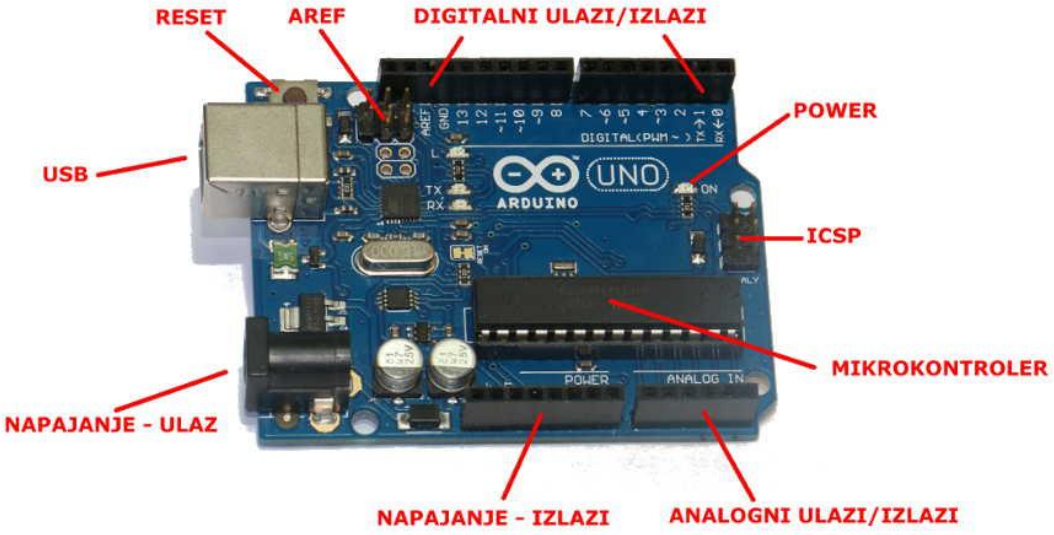
Arduino je instrument za izradu računala koja mogu osjetiti i kontrolirati više iz fizičkog svijeta nego osobno računalo. To je jednostavna mikrokontroler pločica koja je otvorenog koda i razvojno okruženje za pisanje programskog koda za pločicu. *Arduino* programski jezik je implementacija *Wiringa*, slične fizičke platforme, koja je bazirana na procesiranju multimedijalne programske okoline. [1]

Arduino razvojna okolina je dizajnirana da bude jednostavna za korištenje početnicima koji nemaju programskog i elektroničkog iskustva. S *Arduinom* je moguće izraditi objekte koji mogu reagirati i/ili kontrolirati svjetlo, zvuk, dodir i pokret. *Arduino* se koristi za izradu raznolikih stvari uključujući glazbene instrumente, robote, svjetlosne skulpture, igre, interaktivno pokućanstvo (štednjak, hladnjak, mikrovalna pećnica,…) te interaktivnu odjeću. *Arduino* se koristi u mnogim edukacijskim programima širom svijeta, a posebno ga upotrebljavaju dizajneri i umjetnici koji žele na jednostavan način kreirati prototipove, ali nemaju dublje znanje i razumijevanje o tehničkim podacima iza njihovih kreacija.

Arduino je najbolje znan po svojoj opremi (*engl. hardware*), ali je također potrebna i programska podrška (*engl. software*) za programiranje te opreme. I oprema i programska podrška su nazvani *Arduino*. Njihova kombinacija omogućuje kreiranje projekata koja mogu osjetiti i kontrolirati fizički svijet. [2]



Slika 1 Osnovna Arduino pločica



Slika 2 Dijelovi Arduino pločice

Tehničke specifikacije :

- Mikrokontroler ATmega328
- Radni napon : 5V
- Ulazni napon (preporučeni) : 7-12V
- Ulazni napon (maksimalni) : 6-20V
- Digitalni U/I pinovi : 14 (6 pružaju PWM izlaz)
- Analogni izlazni pinovi : 6

- DC struja po U/I pinu : 40 mA
- DC struja za 3.3V pin : 50 mA
- Flash memorija : 32 kB (ATmega328) od kojih se 0.5 koristi za podizanje
- SRAM : 2 kB (ATmega328)
- EEPROM : 1 kB (ATmega328)
- Clock speed : 16 MHz
- Dužina : 68.6 mm
- Širina : 53.4 mm
- Težina : 25g [3]

2.1.1. Povijest *Arduina*

Arduino je započet 2005. godine kao projekt za studente na *Interaction Design Ivrea* u Ivrei, Italiji. U to vrijeme studenti su koristili *BASIC stamp* koji je koštao 100 američkih dolara što je za studente bilo preskupo. Massimo Banzi, jedan od začetnika, predavao je na Ivrei. Ime *Arduino* dolazi od bara u Ivrei, u koji su neki od začetnika projekta zalazili. Bar je nazvan po Arduinu, talijanskom kralju (koji je vladao) od 1002. do 1014. godine.

2.1.2. Arduino komponente

Arduino je proširiv pomoću svojih *shieldova* kojih ima preko 300 od više od 100 proizvođača. Zbog svoje jednostavnosti i jednostavnosti izrade *shieldova* moguće je izraditi vlastitu verziju. Razlog zbog kojeg je *Arduino* popularan su upravo njegovi brojni dodaci i njegova široka primjena. Ne postoji skoro niti jedan slučaj u kojem njegova primjena nije riješena na neki način. *Shieldovi* se mogu slagati jedan na drugog (eng. *stacking*). [19]

Na slici 3 prikazani su *shieldovi* redom:

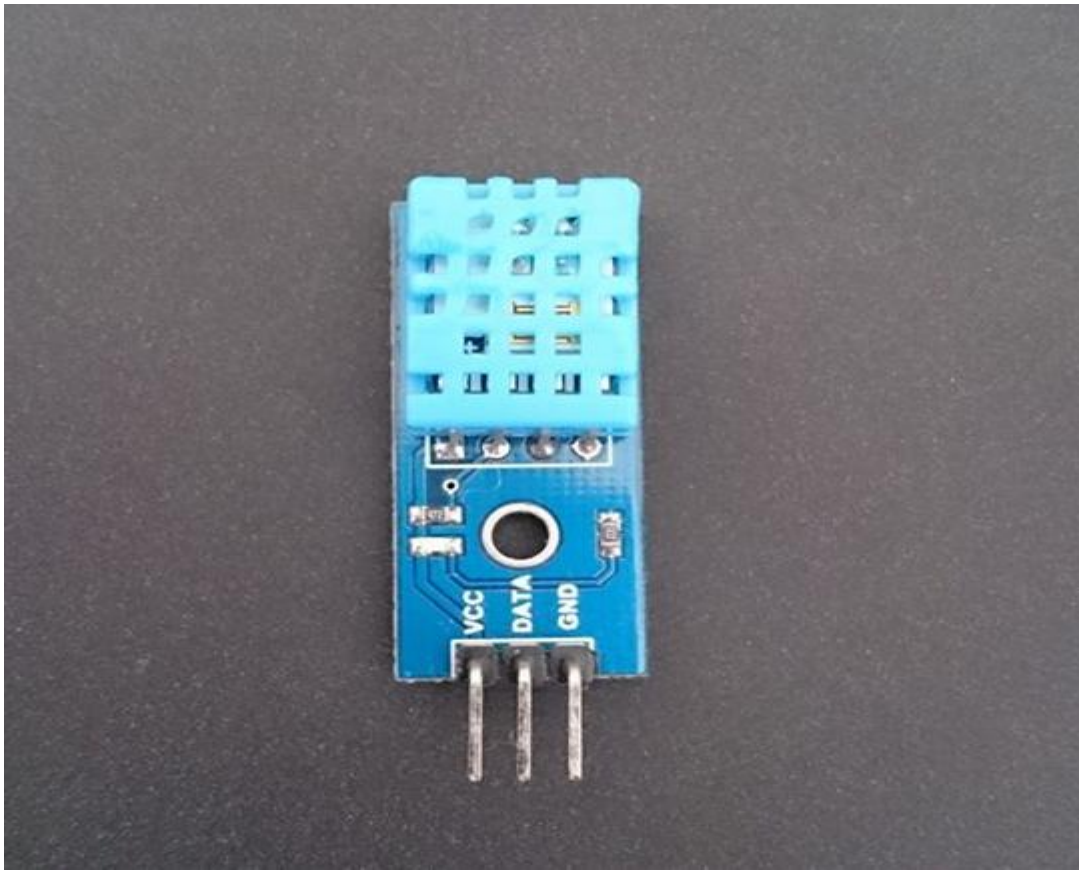
1. ekran Nokije 3310 (nije originalni *shield*)
2. 10BASE-T ethernet *shield*

3. RF primopredajnik
4. senzor za razinu vode
5. LCD ekran 16x2
6. pretvarač USB na TTL Serial
7. ultrazvučni senzor za udaljenost (do 2m)
8. akcelerometar
9. PIR senzor, 10 – IR odašiljač
10. žiro - senzor
11. RF primopredajnik
12. *shield* za SD karticu
13. IR prijemnik
14. RF prijemnik (433MHz)
15. *breadboard* (400 pinova), služi za priključivanje elemenata i *shieldova* na *Arduino*
16. RFID čitač/pisač, 18 – RF prijemnik
17. Numerička tipkovnica

Slika 3 Arduino komponente

2.2. Temperaturni senzor DHT11

DHT11 je senzor za mjerenje temperature i vlage. DHT11 ima 3 pina: GND (eng. ground – uzemljenje), +5V naponski pin, te jedan pin za podatke. Svaki senzor je kalibriran u laboratoriju koji je iznimno precizan u kalibriranju vlage. Kalibracijski koeficijenti su spremljeni u OTP memoriji. Malih dimenzija i male potrošnje, idealan je odabir za razna rješenja uključujući i ona zahtjevnija. Koristi kapacitativni senzor vlage i termootpornik za mjerenje okolnog zraka. Na izlazu daje digitalni signal te mu nije potreban analogni ulazni pin. [7][8][10]

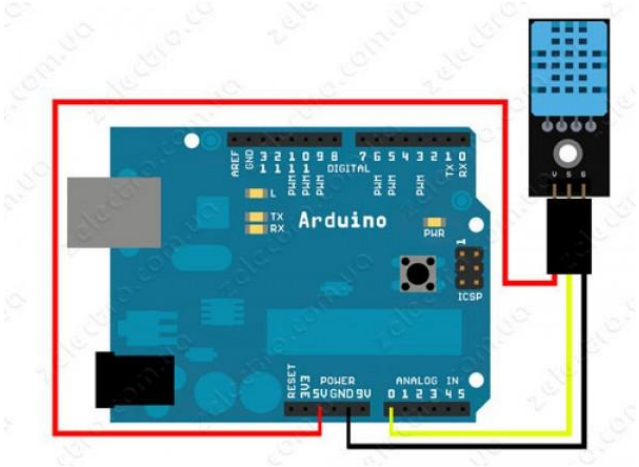


Slika 4 Temperaturni senzor DHT11

Koristi se DHT biblioteka koja vraća vlažnost zraka u postotcima te temperaturu u stupnjevima Celzijusa.

Shema spoja:

DHT11	Arduino
Pin 1	Vcc +5
Pin 2	A2
Pin 4	Gnd



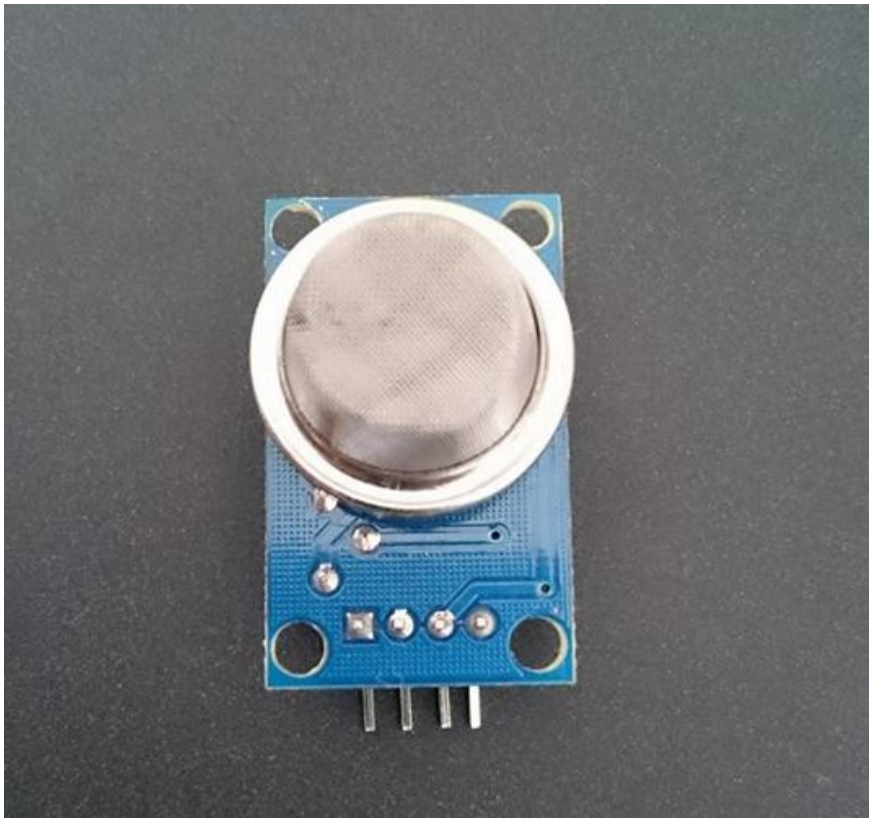
Slika 5 Shema spoja DHT11[9]

Tehničke specifikacije:

- niska cijena
- 3 - 5V power and I/O
- 2.5 mA maksimalna struja
- raspon mjerenja vlažnosti zraka od 20-80% sa 5% pogreške
- raspon mjerenja temperature od 0-50°C sa pogreškom od ±2°C
- dimenzije 15.5mm x 12mm x 5.5mm

2.3. Senzor plinova MQ135

MQ135 je senzor koji se koristi kod općenite kvalitete zraka. Prepoznaje plinove kao što su: amonijak, ugljični dioksid, pare benzena, alkohol, dim,… Koristi mali grijač koji prije mora biti kalibriran tako da se ostavi da kontinuirano mjeri određeni nivo amonijaka u kontroliranim uvjetima. Kako se nisu mogli omogućiti kontrolirani uvjeti i nije bio moguć pristup amonijaku, nije se mogao kalibrirati senzor. Izlaz je analogni signal i može biti čitan sa analognim ulazom *Arduina*.[11][12]



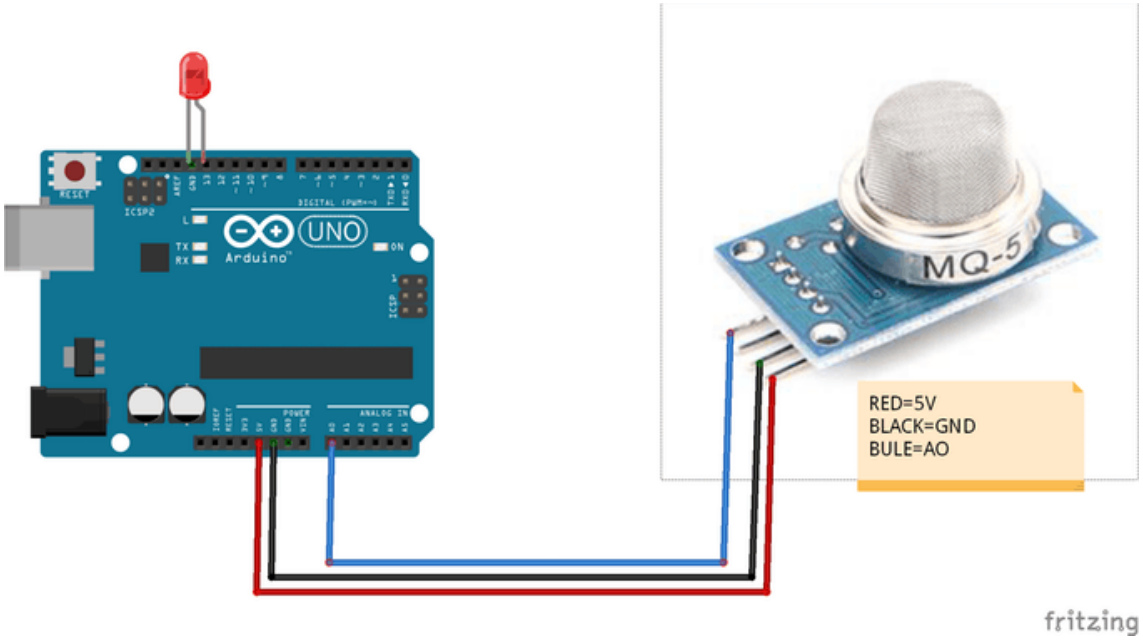
Slika 6 Senzor plinova MQ135

Tehničke specifikacije:

- Niska cijena
- Napon 5V
- Dimenzije 32mm x 22mm x 27mm
- S LM393 komparator
- Digitalni (HIGH/LOW) i analogni (0V-5V) izlaz

Shema spoja:

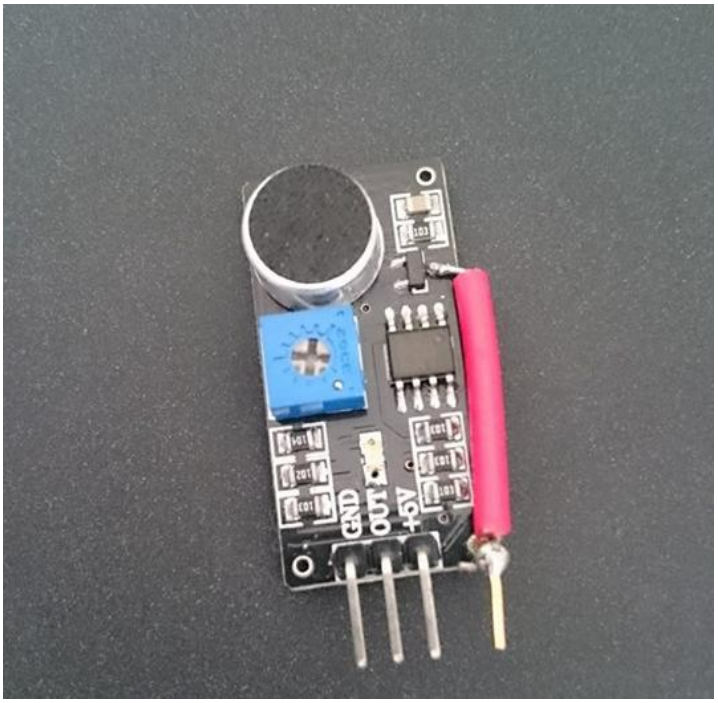
MQ135	Arduino
Pin 1	Vcc +5
Pin 2	GND
Pin 4	A1



Slika 7 Shema spoja MQ135 Arduino

2.4. Modul detektor zvuka

Modul detektor zvuka služi kao senzor za detekciju zvuka u okolini. Prilikom detekcije zvuka dobije se digitalni signal (HIGH/LOW). Osjetljivost se može podesiti potenciometrom na pločici. Kako se htjela dobiti realna veličina zvuka u decibelima, morao se dodati još jedan izlaz (analogni) jer se preko digitalnog samo dobivala informacija ima li ili nema zvuka. Analogni izlaz dodao se između tranzistora i komparatora LM393. Dogodila se slična situacija kao i kod plinskog senzora, modul je trebalo kalibrirati da bi se dobila standardna vrijednost. Kako nije bilo moguće pružiti kontrolirane uvjete i referentni zvuk za kalibraciju, izlaz koji se dobije izražen je u naponskom obliku u rasponu od 0-1023.[13]



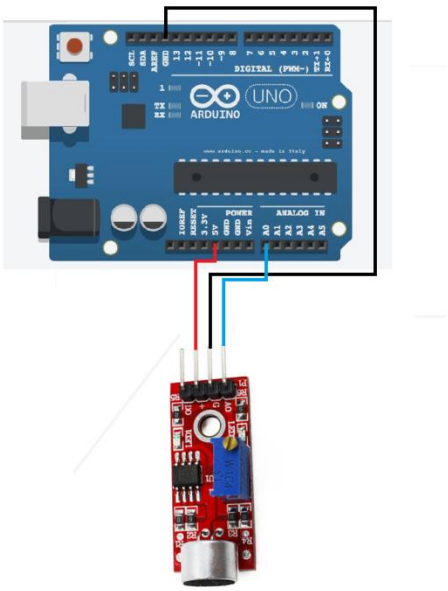
Slika 8 Modul detektor zvuka sa dodanim analognim izlazom

Tehničke specifikacije:

- Niska cijena
- Napon: 4V-6V
- Dimenzije: 38 mm x 16 mm x 10 mm
- Težina: 4 g
- LM 393 komparator

Shema spoja:

Modul	Arduino
Pin 1	GND
Pin 3	Vcc +5
Pin 4	A0

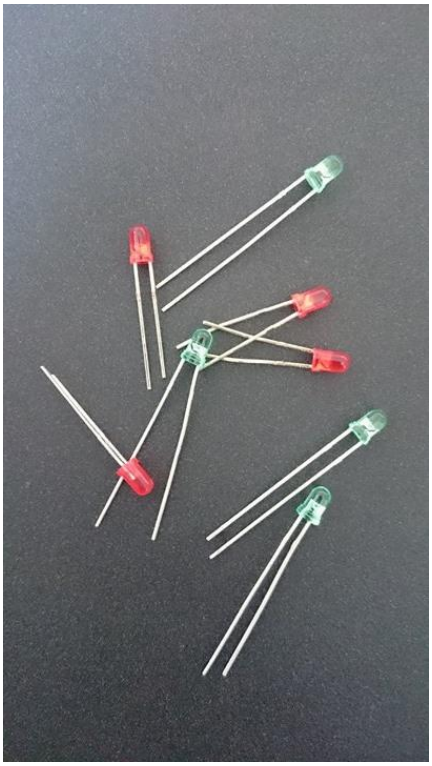


Slika 9 Modul detektor zvuka Arduino shema

2.5. LED diode

Svjetleća dioda ili LED (engl. *light-emitting diode*) je posebna vrsta poluvodičke diode koja emitira svjetlost kada je propusno polarizirana, tj. kada kroz nju teče struja. Kod direktnog miješanja para elektron-šupljina emitira se foton svjetla. Takvo svojstvo imaju poluvodiči galijev arsenid (GaAs), galijev fosfid (GaP) i silicijev karbid (SiC). Ta se pojava naziva elektroluminiscencija. Boja emitiranog svjetla ovisi o poluvodiču, kao i o primjesama u njemu i varira od infracrvenog preko vidljivog do ultraljubičastog dijela spektra. [14]

LED	Arduino
Pin 1	GND
Pin 2	D6-D13



Slika 10 Zelene i crvene diode

Koristilo se osam LED dioda (četiri crvene i četiri zelene) za simuliranje sigurnosnih mjera. Zelene su bile korištene za prikazivanje dozvoljenih vrijednosti, a crvene za

sigurnosno opasne vrijednosti. Ne mogu svijetliti sve diode odjednom. S obzirom da po jednoj vrijednosti senzora ide jedna crvena i jedna zelena dioda, u jednom trenutku po senzoru svijetli ili crvena ili zelena dioda.

2.6. Kućište

Slika 11 Kućište u originalnom obliku

Izrada kućišta nije bila komplicirana, ali do samog kraja nije bilo moguće odrediti na koji način će se rad prezentirati. U početku je postojala ideja da se napravi ploča od pleksiglasa na kojoj će biti pričvrščeni dijelovi, nakon toga ideja je bila da se napravi drveno kućište, ali od toga se ubrzo odustalo zbog teškog oblikovanja i manipuliranja drvetom. Na kraju se najboljim rješenjem pokazalo plastično kućište koja se inače koristi kao razvodna kutija. Dimenzije kućišta su: 190 mm x 140 mm x 75 mm. Zbog većeg broja raznih kućišta različitih dimenzija, mogla se odabrati idealna kutija.

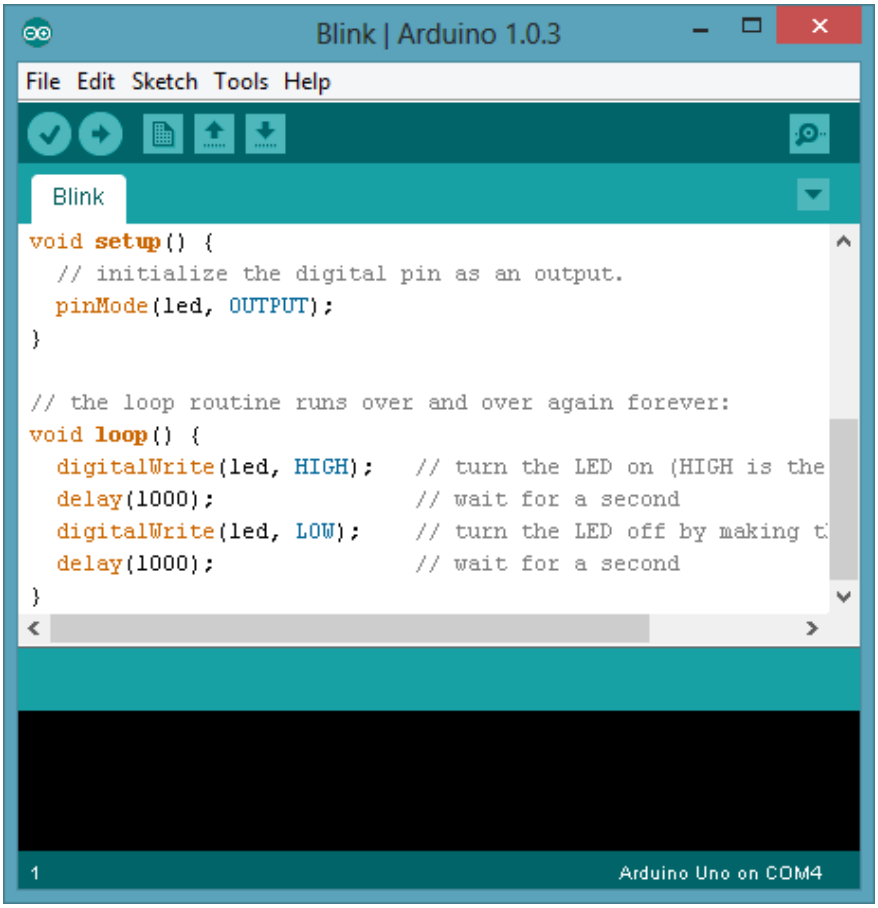
Slika 12 Finalizirani izgled sustava

3.METODE I PROGRAMSKI ALATI

3.1. ARDUINO PROGRAMSKI KOD

Uz *Arduino* dolazi i njegov *software*, besplatnim programskim editorom naredbi u kojem se piše C ili C++ programski kod koji se šalje na *Arduino*. Moguće ga je pokretat na svim glavnim operacijskim sustavima poput: *Microsoft Windows*, *Mac OS* i *Linux*, a može se doći i do izvornog koda. U editoru koda nalaze se brojni primjeri i biblioteke koje služe za lakše snalaženje i pomoć kod izrade vlastitih projekata.

ATmega328 na *Arduino* pločici dolazi sa preprogramiranim *bootloaderom* koji omogućuje uploadanje novog koda na pločicu bez korištenja eksternih programa. Komunicira pomoću STK500 protokola.[19]



Slika 13 Arduino kod

3.1.1. DHT11 Arduino kod

Slika 14 predstavlja kod koji je iskorišten za ispisivanje temperature i vlage na ekran. Prvi dio koda je temperaturni dio u kojem je naredba za ispisivanje izmjerene veličine i petlja kroz koju ta veličina prolazi kako bi se ustanovilo je li dobivena veličina u normalnom rasponu ili nije. Ako je u normalnom uključuje se zelena dioda, ako nije uključuje se crvena. Drugi dio koda je za ispisivanje vlage te se također nalazi petlja koja u ovom slučaju uspoređuje dobivenu veličinu vlage s uvjetima i prema kodu se opet uključuje zelena dioda za dopuštenu vrijednost ili crvena za nedopuštenu.

```
DHT.read11(dht_dpin);
Serial.println(DHT.temperature);
if ((DHT.temperature > 23)&&(DHT.temperature < 35))
  {
    digitalWrite(greenTpin, HIGH);
    digitalWrite(redTpin, LOW);
  }
 else
  {
    digitalWrite(greenTpin, LOW);
    digitalWrite(greenTpin, HIGH);
  }
delay(100);

DHT.read11(dht_dpin);
Serial.println(DHT.humidity);
if ((DHT.humidity > 25)&&(DHT.humidity < 75))
  {
    digitalWrite(greenTpin, HIGH);
    digitalWrite(redHpin, LOW);
  }
 else
  {
    digitalWrite(greenTpin, LOW);
    digitalWrite(greenHpin, HIGH);
  }
delay(100);
```

Slika 14 DHT11 Arduino kod

3.1.2. MQ135 Arduino kod

Slika 15 prikazuje dio programskog koda gdje se definirao analogni pin A1 kao ulazni analogni pin senzora plina i gdje se postavila početna vrijednost senzora plina na nulu.

```
const int plinSensorPin = A1;
int plinSensorValue = 0;
```

Slika 15 Inicijalizacija varijabli i postavljanje vrijednosti

Ostatak koda prikazan na slici 16 prikazuje proceduru plinskog senzora za očitovanje vrijednosti u kojem se prvo definirala varijabla *plinSensorValue* kao analogna vrijednost koja će se dobiti na analognom pinu, a zatim se dobivena vrijednost šalje na serijsku komunikaciju *Arduina* i prikazuje se na ekranu. Sljedeća petlja koristi se za ispitivanje vrijednosti i uspoređivanje sa zadanim uvjetima, a ako je dobivena vrijednost u području normalne vrijednosti uključuje se zelena dioda. U protivnom će se uključiti crvena.

```
// PROCEDURA PLINSKOG SENZORA
// Očitaj vrijoednost sa senzora plina:
plinSensorValue = analogRead(plinSensorPin);
// Pošalji vrijednost serial monitor:
 Serial.println(plinSensorValue);
 if ((plinSensorValue > 20)&&(plinSensorValue < 350))
  {
    digitalWrite(greenTpin, HIGH);
    digitalWrite(redPpin, LOW);
  }
 else
  {
    digitalWrite(greenTpin, LOW);
    digitalWrite(greenPpin, HIGH);
  }
delay(100);
```

Slika 16 Arduino kod senzora MQ135

3.1.3. Modul detektor zvuka Arduino kod

Kao i kod senzora plina MQ135, i kod modula detektora zvuka inicijalizirala se varijabla *soundSensorPin* te se povezala sa analognim ulazom A0, a početna vrijednost se postavila na nulu.

```
const int soundSensorPin = A0;
int soundSensorValue = 0;
```

Slika 17 Inicijalizacija varijabli

U drugom dijelu koda vezanog za modul detektora zvuka, varijabli *soundSensorValue* se dodijelila analogna vrijednost koja će se dobiti na varijabli *soundSensorPin*, odnosno na analognom ulazu A0. Vrijednost koja će se dobiti bit će ispisana pomoću *Arduinove* serijske komunikacije na ekran te će se ujedno i dobivena vrijednost usporediti s uvjetima u petlji koja će odrediti je li vrijednost u granicama dozvoljenog te će se shodno tome uključiti zelena dioda. Ako ona nije u granicama dozvoljenog, uključit će se crvena dioda.

```
// PROCEDURA ZVUČNOG SENZORA
soundSensorValue = analogRead(soundSensorPin);
 // Pošalji vrijednost serial monitor:
  Serial.println(soundSensorValue);
  if (soundSensorValue < 750)
   {
     digitalWrite(greenTpin, HIGH);
     digitalWrite(redZpin, LOW);
   }
  else
   {
     digitalWrite(greenTpin, LOW);
     digitalWrite(greenZpin, HIGH);
   }
 delay(1000);
```

Slika 18 Modul detektor zvuka Arduino kod

3.1.4 Arduino kod – LED diode

Svaka dioda na *Arduinu* ima svoj digitalni izlaz. Jedna nožica LED diode je duža i ona predstavlja plus pol diode, dok kraća predstavlja minus. Na slici 19 je prikazano na koji se pin spajaju plus nožice LED diode.

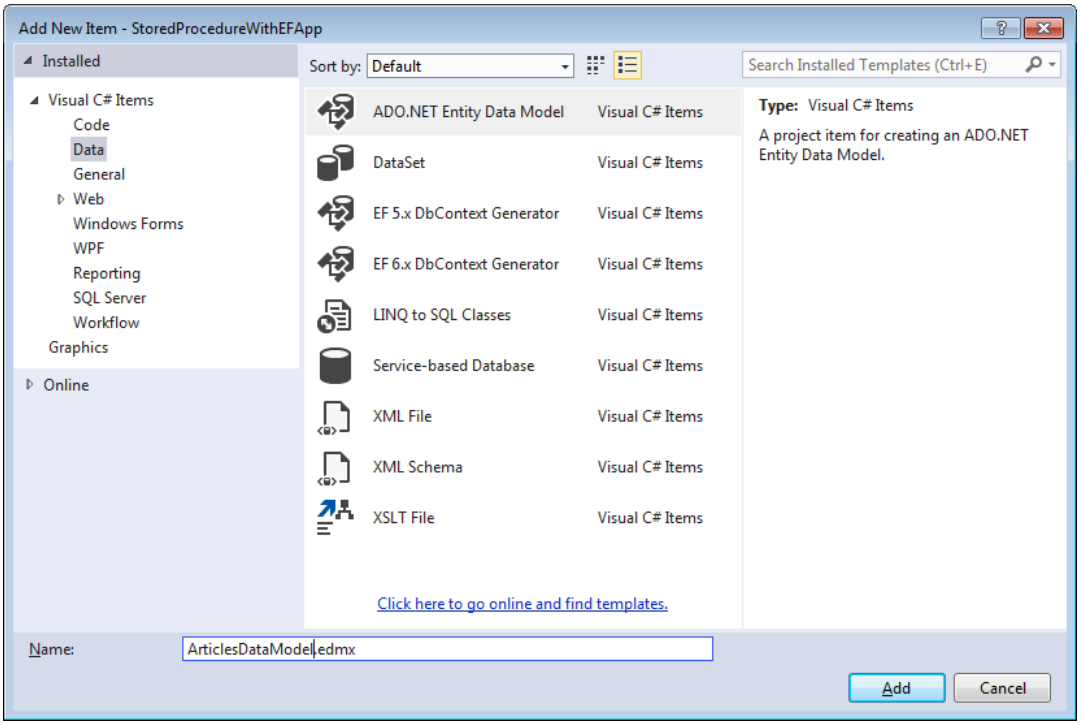
```
int redTpin = 13; //LED crvena dioda za temperaturu
int greenTpin = 12; //LED zelena dioda za temperaturu
int redHpin = 11; //LED crvena dioda za vlagu
int greenHpin = 10; //LED zelena dioda za vlagu
int redPpin = 9; //LED crvena dioda za plin
int greenPpin = 8; //LED zelena dioda za plin
int redZpin = 7; //LED crvena dioda za zvuk
int greenZpin = 6; //LED zelena dioda za zvuk
```

Slika 19 Arduino kod za LED diode

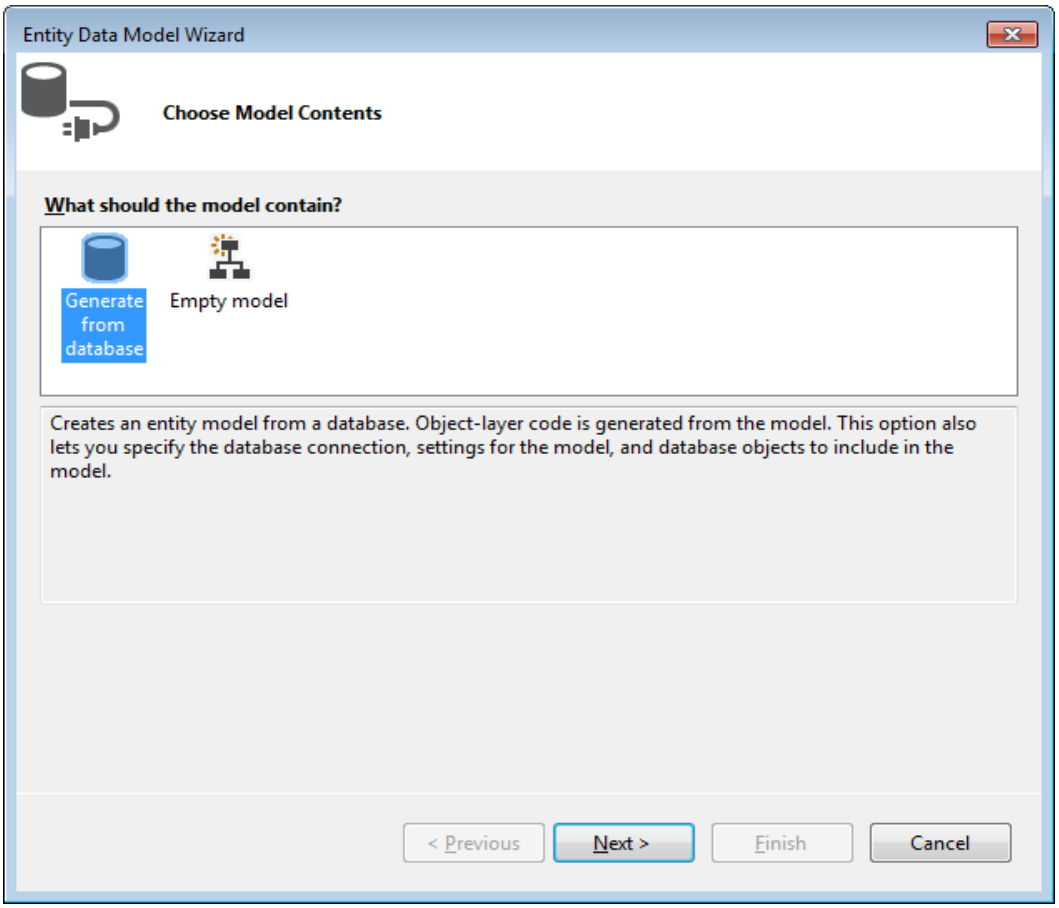
3.2. C# PROGRAMSKI KOD

Za izradu baze podataka u koju će se spremati podaci koji su dobiveni mjerenjem, koristio se *Microsoft Visual Studio* i C# jezik programski alat, točnije *entity framework*. *Entity framework* je objektno-relacijski maper (engl. *mapping*) koji omogućuje .NET

developerima rad sa bazama podataka koristeći .NET objekte. Na taj se način olakšava razvoj podatkovno-orijentacijskih aplikacija jer omogućuje da se podacima bave u formi objekta umjesto da se radi s tablicama i redovima baze podataka. Još jedna velika prednost *Entity Frameworka* je ta što se velik dio koda generira automatski i tako se štedi vrijeme programerima. *Entity Framework* ima nekoliko načina mapiranja, a to su: *database first, code first* i *model first*. Korišten je *database first* model jer iz gotove baze podataka kreira odgovorajuće C# razrede čija svojstva odgovaraju kolonama u tablici po svojstvu i tipu podataka. [20][21][22]



Slika 20. Odabir Entity framework modela



Slika 21. Odabir database first modela

Na sljedećoj slici je prikazan rezultat kreiranja *database first* modela. Kao krajnji rezultat dobilo se nekoliko datoteka koje sadrže opis relacija unutar baze podataka između tablica te njihove odgovarajuće razrede koji su kreirani na temelju dizajna tablica iz baze podataka. EDMX datoteka se koriste za generiranje SQL *server database* shema. Sadrži tri glavne datoteke:

1. *Jezik za definiciju konceptualnih schema (CSDL)*
2. *Jezik za definiciju scheme skladištenja (SSDL)*
3. *Jezik specificiran za mapiranje (MSL)*

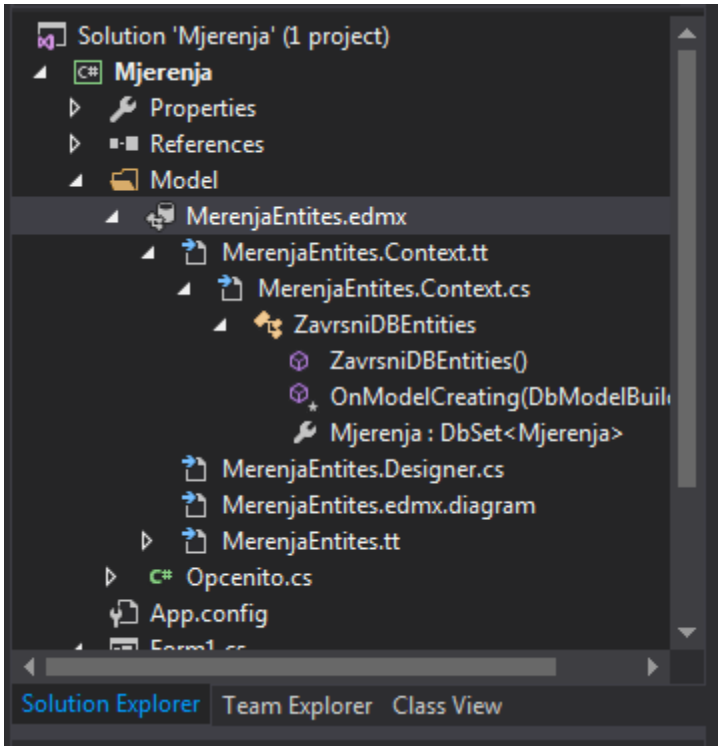
CSDL je XML baziran jezik. Sadrži tip entiteta te svaki tip entiteta ima ime, ključ (za jedinstvene identifikacijske instance) i skup svojstava. Tipovi podataka svojstava su ili jednostavni tipovi poput int ili string ili kompleksni poput razreda i objekta. XML

atributi mogu također sadržavati NULL svojstvo ili početnu vrijednost. Asocijacije definiraju relacije između entiteta.

SSDL je također XML baziran jezik. SSDL sadrži tip entiteta (imena tablica u bazi podataka u ovom slučaju) te svaki tip entiteta sadrži primarni ključ i skup svojstava (stupac tablice baze podataka). Svojstva tipa entiteta sadrže ime, tip stupca i informacije o NULL vrijednostima.

MSL je isto tako XML baziran jezik. MSL je mapa CSDL modela do SSDL modela. *Entity framework* kompajlira CSDL i SSDL glavne datoteke mapiranjem između modela i upita prema bazi zvanih *Client view*. Pogledi se generiraju ili dizajn načinu ili kod pokretanja aplikacije kada se upit prvi puta pozove u CSDL-u. Kad je upit izvršen, njegovi rezultati su pretvoreni i raščlanjeni u naredbeno stablo. Naredbeno stablo predstavlja *umetni*, *ažuriraj*, *obriši* i *odaberi* naredbu.

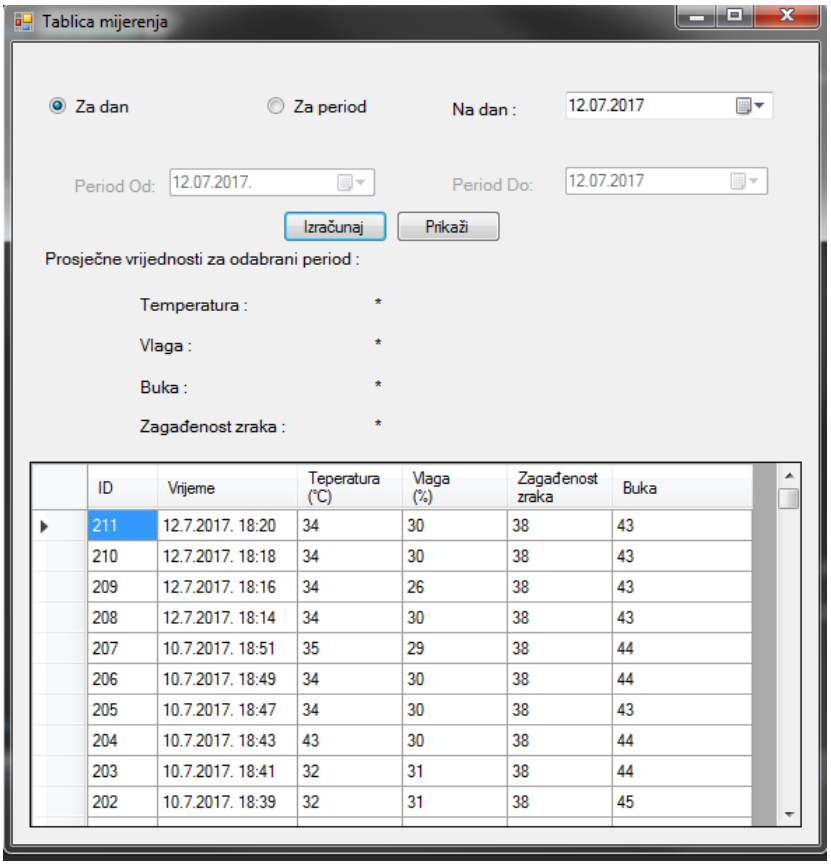
Samo povezivanje aplikacije s bazom podataka se vrši instanciranjem razreda *ZavrsniDBEntities* koji sadrži sve potrebne metode i razrede. [23][24]



Slika 22 Rezultat database first modela

3.2.1. Izgled sučelja aplikacije

Sam izgled aplikacije je prilično jednostavan, sadrži glavni prozorčić na kojem se ispisuju izmjereni podaci pod rednim brojem te datumom i vremenom mjerenja. Ima i gumb za izračunavanje prosječnih vrijednosti za određeni dan ili za neki period vremena te mjesto na kojem se ti podaci prikazuju, kao i gumb za prikazivanje podataka sa istim tim uvjetima.



Slika 23 Izgled aplikacije

3.2.2. Metode

```
public Form1()
{
    InitializeComponent();
    rdbZadaniDan.Checked = true;
    panelPeriod.Enabled = false;
    Arduino.PortName = "COM3";
    Arduino.BaudRate = 9600;
    Arduino.DtrEnable = true;
    rdbZadaniDan.CheckedChanged += new EventHandler(rdbCheckedChange);
    rdbPeriod.CheckedChanged += new EventHandler(rdbCheckedChange);
    System.Timers.Timer aTimer = new System.Timers.Timer();
    aTimer.Elapsed += new ElapsedEventHandler(OnTimedEvent);
    aTimer.Interval = 120000;
    aTimer.Enabled = true;
    populateGrid();
}
```

Slika 24 Prikaz konstruktora glavnog razreda

Na slici 24 je prikazana implementacija konstruktora glavnog razreda u kojem se inicijaliziraju predodređene vrijednosti zadanih dijelova te inicijalizacija različitih događaja vezanih za te dijelove. U sljedećem koraku se točno definira na koji *port* će se poslati podatci koje je izmjerio *Arduino* te kojom će brzinom biti omogućen prijenos podataka. Nakon toga se implementira događaj kod promjene stanja za zadani dan i zadani period te implementacija brojača koji nakon isteka zadanog perioda pokreće metodu za iščitavanje podataka sa Arduina. Nakon što se izvede metoda za iščitavanje podataka sa Arduina, primijenjuje se metoda koja će popuniti aplikaciju dobivenim podatcima.

```
private void DohvatiVrijednosti()
{
    Cursor.Current = Cursors.WaitCursor;
    try
    {
        int temp = 0; int vlaga = 0; int gas = 0; int buka = 0;
        Thread.Sleep(5000);
        if (Arduino.BytesToRead != 0)
        {
            var content = Arduino.ReadExisting();
            Arduino.DiscardInBuffer();
            Arduino.DiscardOutBuffer();

            string[] data = content.ToString().Split('\r');
            if (Int32.TryParse(data[0].Trim().Substring(0, 2), out temp))
                temp = Convert.ToInt32(data[0].Trim().Substring(0, 2));
            if (Int32.TryParse(data[1].Trim().Substring(0, 2), out vlaga))
                vlaga = Convert.ToInt32(data[1].Trim().Substring(0, 2));
            if (Int32.TryParse(data[2].Trim().Substring(0, 2), out gas))
                gas = Convert.ToInt32(data[2].Trim());
            if (Int32.TryParse(data[3].Trim().Substring(0, 2), out buka))
                buka = Convert.ToInt32(data[3].Trim());

            Model.Mjerenja mjerenja = new Model.Mjerenja();
            mjerenja.Mje_Temperatura = temp;
            mjerenja.Mje_Vlaga = vlaga;
            mjerenja.Mje_Id = getMaxID();
            mjerenja.Mje_Vrijeme = DateTime.Now;
            mjerenja.Mje_ZagadjenostZraka = gas;
            mjerenja.Mje_Buka = buka;

            SpremiPodatke(mjerenja);
        }
    }
```

Slika 25 Metoda DohvatiVrijednosti

U metodi *DohvatiVrijednosti* se prvo deklarira i inicijalizira varijabla svakog senzora te se postavlja na nulu. Nakon toga slijedi provjera nalaze li se na serijskom portu na koji je povezan *Arduino* bilo kakvi podatci. U sljedećim koracima se iščitavaju podaci sa serijskog porta te se čisti. Postavlja se nekoliko uvjeta pomoću kojih se provjerava mogu li se dolazni podaci pretvoriti iz znakovnog niza u cijeli broj. Na kraju metode *DohvatiVrijednosti* podaci se prebacuju iz ranije deklariranih varijabli u odgovarajući model koji se dobiva generiranjem iz baze podataka. Posljednjom naredbom se otvara konekcija prema bazi podataka te se u nju spremaju dobiveni rezultati.

```
private void populateGrid()
{
    using (ZavrsniDBEntities db = new ZavrsniDBEntities())
    {

        var queryOpcenito = from m in db.Mjerenja
                            select new Opcenito
                            {
                                ID = m.Mje_Id,
                                Teperatura = m.Mje_Temperatura,
                                Vrijeme = m.Mje_Vrijeme,
                                Buka = m.Mje_Buka,
                                Vlaga = m.Mje_Vlaga,
                                ZagađenostZraka = m.Mje_ZagadjenostZraka
                            };
        if (queryOpcenito != null)
            if (queryOpcenito.Count() < 1)
                opcenitoBindingSource.DataSource = queryOpcenito.OrderByDescending(x => x.ID).ToList();
            else
                opcenitoBindingSource.DataSource = queryOpcenito.OrderByDescending(x => x.ID).ToList();

    }
}
```

Slika 26 Popunjavanje aplikacije

Metodom *populateGrid* se otvara konekcija prema bazi te se piše upit pomoću LINQ načina pisanja koji omogućuje da se automatski popuni lista koju čine objekti tipa *općenito* te se isti prije toga pune podatcima iz baze. Sljedećim uvjetom provjerava se sadrži li dobivena lista podatke. Zatim se unutar te liste sortiraju podaci na temelju vrijednosti ID svojstva te se takva sortirana lista dodjeljuje izvoru podataka od tablice koja je prikazana na zaslonu aplikacije.

```
private void SpremiPodatke(Model.Mjerenja model)
{
    try
    {
        using (ZavrsniDBEntities db = new ZavrsniDBEntities())
        {
            Application.DoEvents();
            db.Mjerenja.Add(model);
            db.SaveChanges();
        }
    }
    catch (Exception e)
    {
        MessageBox.Show("Pogreška prilikom spremanja podataka!" + Environment.NewLine + e.Message, "", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }

}
```

Slika 27 Metoda spremi podatke

Metodom *SpremiPodatke* se otvara konekcija prema bazi podataka. U model koji je izgenerirao *Entity Framework* pridodaje se model istog tipa koji se prije napunio u metodi *DohvatiVrijednosti*. Zatim se poziva metoda *SaveChanges* koja je dio generiranog modela koja sprema sve promjene unutar objekta *db*, što znači da postoji slučaj s više tablica da se u jednu upisuju dok se iz drugih brišu zapisi. Dovoljno je samo jedanput pozvati metodu *SaveChanges* kako bi se spremile promjene. Implementira se i upravljanje greškama na način da se, ako dođe do greške, korisniku otvara dijalog s porukom prikazanom na slici 27.

4. ZAKLJUČAK

Arduino je elektronička pločica otvorenog koda (engl. o*pen source*) koja se temelji na jednostavnom, ali dovoljno jakom i naprednom hardveru i softveru za neka jednostavna ili kompliciranija rješenja. Zbog svojih mogućnosti ima široku primjenu unutar računarske struke, ali i izvan nje. Tako zbog svoje jednostavnosti može biti upotrebljena kao baza za pametnu kuću, privatni sigurnosni sustav, autić na daljinsko upravljanje, parking senzor ili, kao u ovom slučaju, sustav za sigurnost od raznih vanjskih utjecaja u nekom radnom prostoru ili postrojenju.

Ono što se ovdje pokušalo napraviti je minijaturna replika većih i mnogo skupljih sustava za nadzor i upravljanje nad prostorima i postrojenjima. Jedan od sustava koji se pokušao replicirati je SCADA (engl. s*upervisory control and data acquisition*) sustav koji je u potpunosti automatiziran i ima kontrolu nad postrojenjima. Iako sustav nije u potpunosti repliciran, utvrdilo se da to nije nemoguće ukoliko se uloži mnogo vremena, truda i novca. S obzirom na okolnosti i mogućnosti, ovaj je završni rad dovršen i zadovoljava osnovne uvjete svakog profesionalnog sustava za nadzor i kontrolu u radnim prostorima i postrojenjima, ali ne može konkurirati ostalima bez obzira što bi za poluprofesionalnu upotrebu bio dostatan. Ovaj sustav je minimalno deseterostruko jeftiniji od gotovih profesionalnih sustava, a dovoljno je precizan, brz i jednostavan za korištenje da zadovolji krajnjeg korisnika. Prednost ovog završnog rada je ta što ima mogućnost izrade prema potrebama korisnika te može imati još veće mogućnosti. Ipak, tu postoje i neki nedostaci kao što su senzori koji se moraju kalibrirati u uvjetima koje je u domaćoj radnji nemoguće dobiti te činjenica da što ima više modula i senzora, to je završni rad zbog žica kompliciraniji te je teže zamijeniti neki dio ako se pokvari.

Na ovaj završni rad potrošeno je otprilike 700 kn i oko godinu dana rada, neki dijelovi su se naknadno dodavali, neki odstranjivali, neki su bili u planu, ali nisu realizirani zbog vanjskih utjecaja. Tako je u jednom trenutku postojala želja da se završni rad spoji bežično s bazom podataka pomoću *Bluetooth* modula HC-06. Nažalost, dolazilo je do greške u serijskom spajanju gdje je serijski port bio u stanju zauzeća i to u trenutku kada su se podaci trebali poslati preko *Bluetootha* na serijski port. Razlog zbog kojeg se to

događalo nije pronađen jer svaka metoda i rješenje koje je isprobano je javljalo istu grešku te se zbog toga ta ideja napustila. Postoji želja da će se s vremenom ovaj završni rad još više unaprijediti i približiti ideji potpuno automatiziranog sustava za upravljanje i praćenje.

5. POPIS KOMPONENTI I CIJENE

KOMPONENTA	~CIJENA (HRK)
Arduino Uno	280,00
Testna pločica	35,00
Žice (M-M)	25,00
Žice (M-Ž)	25,00
Konektor napajanja	20,00
DHT11	20,00
MQ135	39,00
Modul detektor zvuka	29,00
HC-06	99,00
LED diode	10,00
Kućište	32,00
Ležište za diode	40,00
Baterija 9V	36,00
UKUPNO	690,00

6. POPIS SLIKA

7. LITERATURA

[1] http://arduino.cc/en/Guide/Introduction

[2] Arduino Cookbook, 2nd Edition (Michael Margolis, 2011)

[3] http://arduino.cc/en/main/arduinoBoardUno

[4] http://www.amci.com/tutorials/tutorials-what-is-programmable-logic-controller.asp

[5]http://www.allaboutcircuits.com/textbook/digital/chpt-6/programmable-logic-controllers-plc/

[6] https://inductiveautomation.com/what-is-scada

[7] https://playground.arduino.cc/Main/DHT11Lib

[8] http://www.micropik.com/PDF/dht11.pdf

[9] http://www.sivava.com/image/cache/data/products/arduino/k05-arduino/500-k05-arduino-1-600x600.jpg

[10] http://robocraft.ru/files/datasheet/DHT11.pdf

[11] https://e-radionica.com/hr/mq135-senzor-plinova.html

[12]https://www.olimex.com/Products/Components/Sensors/SNS-MQ135/resources/SNS-MQ135.pdf

[13] https://e-radionica.com/hr/blog/2015/08/19/kkm-modul-detektor-zvuka-2/

[14] https://www.fkit.unizg.hr/_download/repository/VP11.pdf

[15] https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf

[16]https://www.banggood.com/HC-06-Wireless-Bluetooth-Transceiver-RF-Main-Module-Serial-For-Arduino-p-80364.html

[17] https://e-radionica.com/wp/hrvatski/wp-content/uploads/sites/3/2016/03/Untitled-Sketch_bb1.png

[18] https://e-radionica.com/hr/eksperimentalna-plocica.html

[19] https://sysportal.carnet.hr/node/1418

[20]https://www.helloworld.rs/blog/ENTITY-FRAMEWORK-za-data-orijentisane-aplikacije/289

[21] https://docs.microsoft.com/en-us/ef/

[22]http://www.c-sharpcorner.com/UploadFile/abhikumarvatsa/database-first-approach-in-entity-framework/

[23] https://fileinfo.com/extension/edmx

[24] http://www.c-sharpcorner.com/UploadFile/ff2f08/entity-framework-part-1/